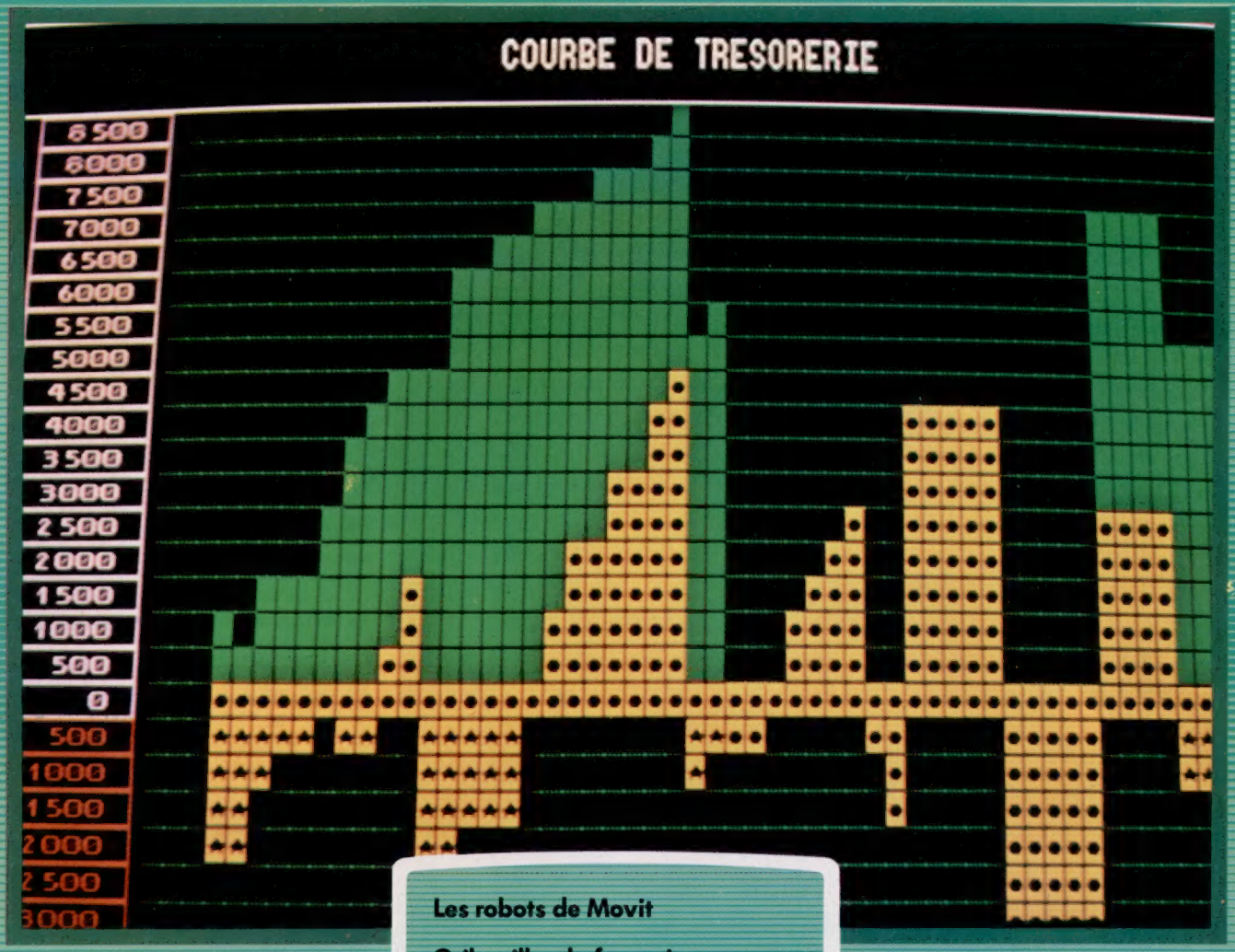
ABC INFORMATIQUE

N° 79

COURS D'INFORMATIQUE PRATIQUE ET FAMILIALE



Les robots de Movit

Gribouille, du français sur Apple

Grand prix en MSX

Organisation des données

EDITIONS ATLAS

M 6062-79-12,50 F 90 FB - 3,80 FS - $ 1.95

Page manquante
(publicité et colophon)

Pouvoir aux élèves

Une tortue est un petit robot mécanique très simple, qui peut favoriser la compréhension par les écoliers de concepts mathématiques abstraits. Une façon d'apprendre en s'amusant !

Mieux qu'un long discours
Ces enfants abordent la géométrie grâce à une tortue. Ils apprennent à la programmer pour qu'elle puisse se déplacer sans encombre dans un labyrinthe qu'ils ont créé eux-mêmes. Ils se familiarisent ainsi avec des notions aussi abstraites que le déplacement angulaire, concept difficile auquel ils ont d'habitude le plus grand mal à s'intéresser.
(Cl. Tony Sleep.)

L'emploi de robots en pédagogie permettra peut-être de surmonter un problème difficile : les jeunes enfants ont du mal à saisir des concepts abstraits sans l'appui d'exemples très concrets. Certaines notions de trigonométrie, par exemple, peuvent être matérialisées à l'aide de la tortue, dont les mouvements illustrent parfaitement certains processus de réflexion. Ce peut être aussi un bon moyen de s'initier à la programmation.

La géométrie est traditionnellement enseignée à l'aide de règle, de crayon, de rapporteur et de compas. Mais, pourquoi un enfant de huit ans éprouverait-il le besoin de mesurer un angle, ou même tout simplement de s'intéresser à la question? D'un autre côté, il est plus attrayant de savoir que la détermination d'un angle amènera, ou non, la tortue à heurter les bouteilles en plastique qui, posées sur le sol, symbolisent une forêt. De la sorte, même si la géométrie elle-même peut rester une notion un peu lointaine, il est au moins possible de motiver, d'intéresser l'enfant à des applications pratiques; la différence entre 15 cm et 20 cm devient vite évidente après qu'on a choisi entre un angle de 5° et un autre de 8° par exemple. Et il y en a de nombreux.

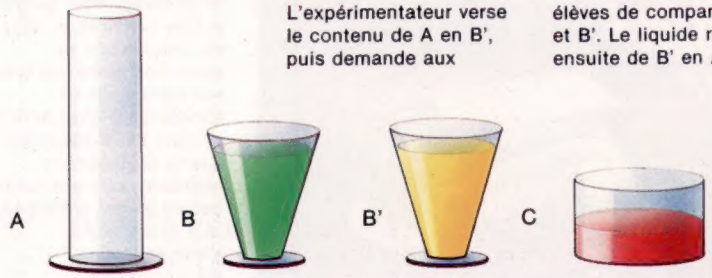
La tortue offre aussi l'avantage de permettre un apprentissage par la pratique des essais et erreurs : l'enfant peut modifier son programme jusqu'à ce qu'il lui donne satisfaction, et il apprendra bien des choses au passage sur les angles, les lignes et les méthodes de mesure. Une erreur signifie simplement qu'il faut faire quelque chose — et non que tout était faux. Programmer un bras mécanique pour qu'il puisse soulever et déplacer un objet oblige à penser de façon tridimensionnelle. Une simple analyse sur le papier ne pourrait suffire; l'expérience donne de bien meilleurs résultats. On voit nettement les mouvements de l'appareil, toutes les erreurs de programmation deviennent évidentes, et il est aisé de procéder à des modifications. A chaque instant, le programmeur en herbe peut expérimenter et avoir aussitôt les résultats de son travail, qui devient ainsi nettement plus personnalisé.

Bien des écoles primaires anglaises se sont dotées du Big Trak, un véhicule-jouet d'allure futuriste qui dispose d'une petite mémoire interne et d'un clavier numérique. Des touches fléchées permettent de lui donner une direction et cette commande est suivie d'un nombre compris entre

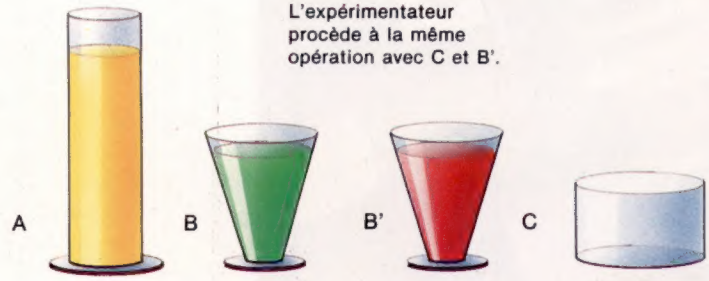
Un exemple concret

Les jeunes enfants ont toujours du mal à saisir les concepts abstraits, même illustrés par des exemples. Bien connu est le cas de la conservation des quantités et des transitions relatives : l'élève se voit présenter des récipients de formes différentes, mais de capacité égale. A, B et C contiennent autant l'un que l'autre, et B et B' sont identiques. Le liquide passe d'un récipient à l'autre, et on demande à l'enfant quel sera le résultat.

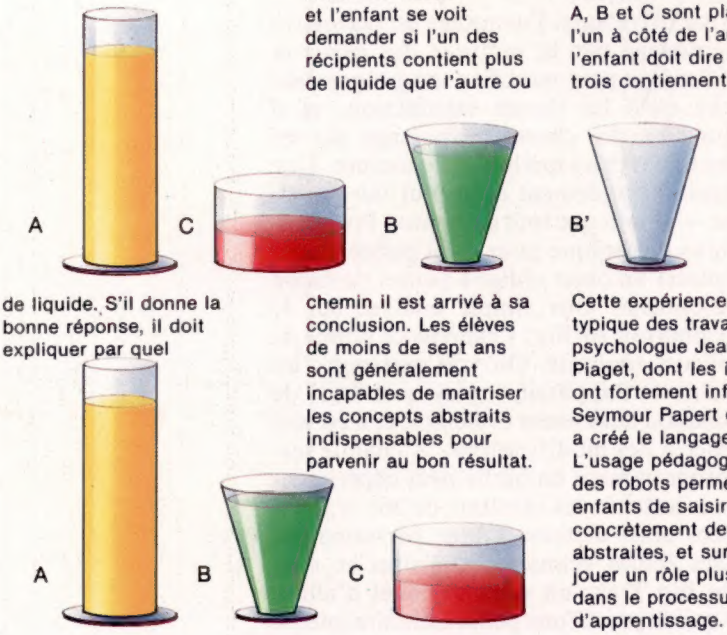
L'expérimentateur verse le contenu de A en B', puis demande aux élèves de comparer B et B'. Le liquide repasse ensuite de B' en A.



L'expérimentateur procède à la même opération avec C et B'.



A et C sont placés l'un à côté de l'autre et l'enfant se voit demander si l'un des récipients contient plus de liquide que l'autre ou si les deux en contiennent autant. A, B et C sont placés l'un à côté de l'autre et l'enfant doit dire si tous trois contiennent autant de liquide. S'il donne la bonne réponse, il doit expliquer par quel chemin il est arrivé à sa conclusion. Les élèves de moins de sept ans sont généralement incapables de maîtriser les concepts abstraits indispensables pour parvenir au bon résultat.



Cette expérience est typique des travaux du psychologue Jean Piaget, dont les idées ont fortement influencé Seymour Papert quand il a créé le langage LOGO. L'usage pédagogique des robots permet aux enfants de saisir concrètement des idées abstraites, et surtout de jouer un rôle plus actif dans le processus d'apprentissage.

Kevin Jones

1 et 99. Par exemple, AVANCE 1 fait avancer l'engin droit devant lui, d'une distance égale à sa propre longueur. DROITE 15 le fera tourner sur la droite de « 15 min », soit 45°. La mémoire du Big Trak peut gérer jusqu'à huit commandes à la fois. C'est une sorte de substitut de la tortue, qui permet aux enfants certains exercices de programmation très simples.

Big Trak fait la démonstration que des commandes sont exécutées dans le même ordre dans lequel elles ont été données. N'est-ce pas une intéressante initiation à la programmation des ordinateurs ?

De quoi penser

Toutefois, si les tortues peuvent jouer un rôle dans l'enseignement des mathématiques, ce n'est pas là leur but premier : leur créateur, Seymour Papert, voulait initier les enfants à la programmation sur ordinateur. Son idée de base était qu'ainsi ils pourraient programmer l'appareil — et non l'inverse. Par la même occasion, Papert met au point le langage LOGO : la tortue devient « un objet avec lequel on pense », tandis qu'elle est guidée par les écoliers eux-mêmes, grâce au LOGO. Ils s'initient de cette façon à la programmation.

Paul Cheung, de l'université d'Édimbourg, s'est tout particulièrement intéressé à l'aspect robotique du LOGO : à une époque où l'on se préoccupait avant tout du matériel, il a cherché à améliorer la qualité des logiciels. Les enfants devaient pouvoir contrôler toutes sortes de robots.

Le LOGO est parfait pour la programmation des tortues, mais il est beaucoup plus limité dès qu'il lui faut gérer d'autres engins. Cheung mit donc au point une version plus structurée du langage, le C-LOGO (Concurrent-LOGO), aux fonctions plus nombreuses et capable d'accueillir des entrées en provenance de capteurs sensoriels, de lecteurs de codes-barres, de sondes lumineuses. Le C-LOGO est capable de « traitement parallèle » (c'est-à-dire qu'il peut exécuter simultanément plusieurs tâches).

La maison robotisée

Il contrôle ainsi jusqu'à huit fonctions en même temps. Certaines d'entre elles sont particulièrement intéressantes : CHAQUE FOIS QUE lance une action que C-LOGO mène ensuite à bien ; LORSQUE attend qu'une condition soit satisfaite pour activer telle ou telle tâche, ce qui est essentiel pour un traitement en parallèle. Il est aussi possible d'allumer ou d'éteindre des commutateurs, et de savoir quel est l'état de chacun d'eux. Cela permet l'emploi de capteurs qui détectent les collisions et informent l'ordinateur dès qu'un contact est enregistré. Les enfants programmeront leur robot en faisant usage de la commande LORSQUE à chaque rencontre avec un obstacle et lui commanderont de changer de direction.

C-LOGO demeure évidemment plus adapté aux enfants en âge d'aller au lycée (Cheung a fait de

nombreuses expériences dans ce cadre). Des écoliers de treize à quinze ans ont ainsi rédigé des programmes pour faire tourner des moulins à vent, déplacer des ascenseurs, des camionnettes et des bras mécaniques, contrôler les portes et les fenêtres dans une maison robotisée. Ce dernier exemple est particulièrement intéressant : l'ouverture d'une fenêtre déclenchait un système d'alarme, et les portes ne pouvaient être ouvertes qu'après entrée au clavier d'un mot de passe. Il s'agissait en fait de deux programmes différents, lancés simultanément grâce au traitement parallèle.

La maison comportait trois étages, tous accessibles par ascenseur. Celui-ci était construit avec des pièces de Meccano et alimenté par un moteur à courant continu pourvu de commutateurs permettant de savoir à quel étage on était. L'enfant apprenait très vite à appuyer sur tel ou tel bouton pour l'envoyer au rez-de-chaussée, au premier ou au deuxième. Cela les amenait à poser la question : « Qu'est-ce qui se passe si on appuie sur un bouton alors que l'ascenseur est entre deux étages? »

Leur première réaction fut de programmer l'ensemble afin qu'il ignore toute intervention extérieure tant qu'il ne serait pas arrivé à destination. En y réfléchissant, toutefois, ils virent qu'il serait bien préférable de doter l'ordinateur de la capacité de se souvenir des instructions, puis de les mettre en œuvre selon des règles bien définies à l'avance.

Des solutions simples

C-LOGO leur permit de rédiger, sans trop de difficultés, un programme qui, autrement, se serait révélé bien plus complexe. Ils avaient besoin d'un programmeur, d'un détecteur de signal et d'un contrôleur de l'ascenseur (ces deux derniers travaillant en parallèle). CHAQUE FOIS QUE rendait possible la prise en compte des signaux, transmis ensuite au programmateur. Le contrôleur lui demandait alors une destination; cela étant fait, il en réclamait une autre.

Une fois la séquence des mouvements déterminée, elle était exécutée. Les enfants avaient sans cesse sous les yeux le dispositif et les moyens de le contrôler : ils persévérèrent jusqu'à trouver la bonne solution.

Bien des pédagogues pensent en effet que, grâce aux robots, les écoliers peuvent « voir » les processus de réflexion et venir à bout des blocages. Procédant par essais et erreurs, contraints de penser de manière séquentielle, ils ne réfléchissent plus en termes de « vrai ou faux », mais abordent véritablement un environnement changeant, réaliste, dans lequel il faut recourir sans cesse à des modifications pour s'adapter.

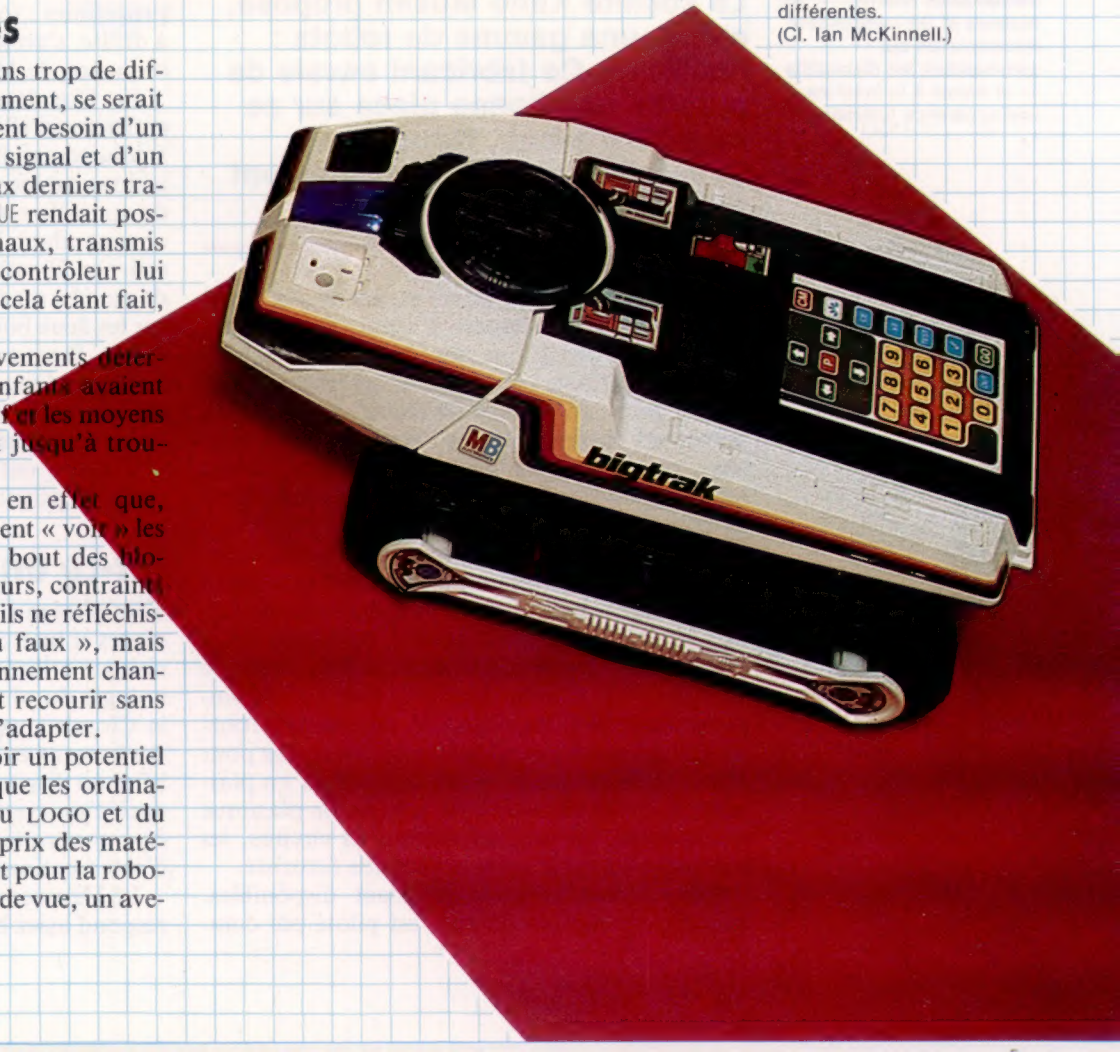
Les robots pourraient donc avoir un potentiel éducatif encore plus important que les ordinateurs eux-mêmes. La création du LOGO et du C-LOGO, la baisse constante des prix des matériels et l'intérêt sans cesse croissant pour la robotique laissent présager, de ce point de vue, un avenir prometteur.

Jouer avec la tortue

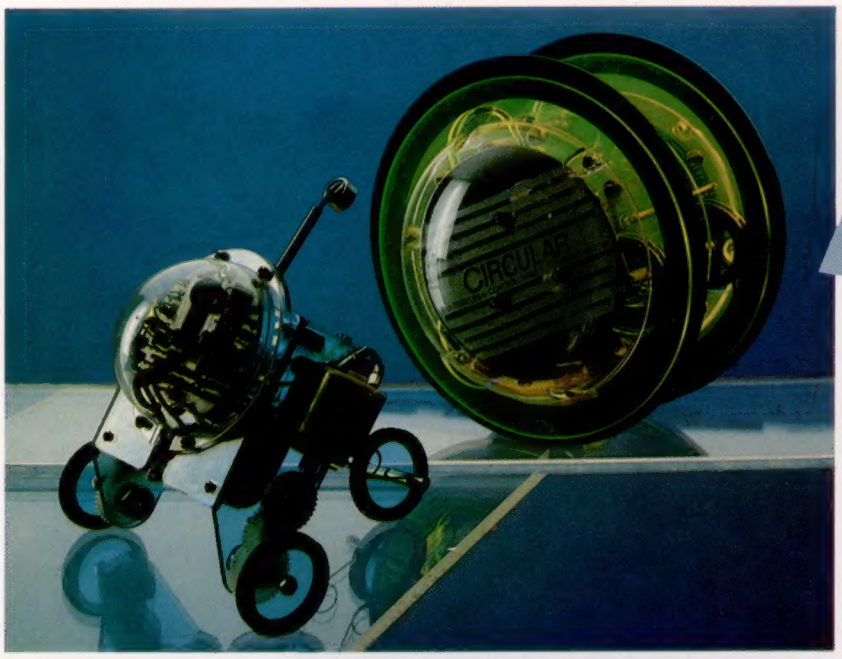
L'usage de la tortue en salle de classe n'est pas limité à la géométrie, la trigonométrie ou la programmation. Un labyrinthe peut ainsi être intégré au sein d'un projet sur la mythologie (Thésée et le Minotaure) ou sur l'architecture (le labyrinthe du dallage de la cathédrale de Reims). Un projet encore plus ambitieux a abouti à la création d'une ville entière; la tortue devait passer d'un magasin à l'autre, afin d'y faire des achats portés sur une liste.

D'autres idées à citer : une course (chaque enfant doit programmer la tortue de sorte qu'elle parcoure un trajet défini; c'est celui qui lui a fait accomplir le trajet le plus rapide qui gagne) ou l'envoi de messages; (les enfants s'assoient en cercle et se servent des tortues pour communiquer entre eux).

Le jeu n'est pas non plus à négliger. Les écoliers peuvent, par exemple, placer la tortue au centre d'un cercle divisé en plusieurs sections. Chacune porte un nombre différent : la tortue doit les parcourir, puis venir s'arrêter sur l'une d'elles (c'est une sorte de variante du jeu de Simon). Le vainqueur est celui qui fait le plus grand nombre de points en un minimum de mouvements.



Chaud devant
Big Trak est un engin autonome qui peut être programmé de façon analogue au LOGO grâce à son clavier numérique. On entre une direction suivie d'un nombre (jusqu'à 99). Big Trak peut accomplir de manière séquentielle jusqu'à huit commandes différentes.
(Cl. Ian McKinnell.)



En mouvement
Nous voyons ici les trois robots les plus coûteux de la gamme Movit. De gauche à droite, nous retrouvons le Piper Mouse, le Circular et le Memocon Crawler. L'extérieur de chaque machine, en plastique transparent, laisse apparaître les mécanismes internes — les moteurs et les composants électroniques qui commandent les dispositifs — et donne à la machine une apparence futuriste.

Les robots Movit

La société Kaho Musen propose, en kit, une gamme de robots : les Movit. Ce fabricant essaie de prendre une bonne place sur ce marché d'avenir. Nous allons déterminer si ces machines sont véritablement des robots.

Le niveau de développement atteint actuellement en robotique se compare à celui où se situait la micro-informatique au début des années soixante-dix. Alors que le micro-ordinateur était mis au point, dans un secret relatif, par quelques personnes passionnées par l'électronique, beaucoup plus de monde est aujourd'hui conscient qu'une nouvelle révolution micro-électronique est en cours.

Au vu de l'intérêt que suscite cette technologie, il n'est pas étonnant qu'un fabricant lance une série de robots domestiques peu coûteux. Les Movit, fabriqués par la société japonaise Kaho Musen, sont commercialisés à des prix allant de 150 F à 500 F. Chaque Movit est livré en kit complet (seules les piles, de 1,5 V, ne sont pas fournies). Le kit comprend même un tournevis pour effectuer le montage ! Avec leur boîtier en plastique robuste et avec une sphère en plastique transparent, laissant apparaître les circuits, les machines ont une belle apparence futuriste.

Cinq Movit sont actuellement disponibles. Monkey, le moins coûteux, est piloté par commande sonore : sollicité par l'intermédiaire du capteur sonore — en « entendant » par exemple un claquement de mains ou un sifflement —, ses deux bras à mouvement alternatif tirent le dispositif le long d'une corde ou d'un autre objet long et étroit. Le capteur est un microphone à condensateur connecté à une carte de circuits imprimés qui décode le signal. Des signaux appropriés démarrent le moteur électrique CC qui, à son tour, actionne le mécanisme et fait bouger les bras. Un chronomètre préréglé sur la carte assure l'arrêt du Monkey.

Le deuxième Movit dans la gamme de prix est le Line Tracer II. Ce robot, commandé par un capteur infrarouge placé sur sa base, suit une ligne dessinée au sol. Ce robot est mû par deux moteurs CC qui actionnent ses trois roues.

Légèrement plus coûteux, le Piper Mouse est commandé par un capteur sonore qui répond, lui, à un sifflet de chien. A chaque coup de sifflet, le Movit exécute l'une des séquences de commandes prédéfinies : tourner à gauche, s'arrêter, tourner à droite, s'arrêter, avancer, s'arrêter. Après cette séquence, le Movit se déplace en ligne droite.

Le Circular Movit est la première machine entièrement télécommandée. Il est commandé par fréquences radio et mû par deux moteurs CC qui actionnent les grosses roues extérieures. La boîte de contrôle est munie de deux boutons, chacun commandant l'un des moteurs. Pour le faire tourner, il suffit donc d'appuyer sur le bouton de gauche ou de droite ; appuyer simultanément sur les deux boutons fait avancer le robot en ligne droite. Deux cartes de circuits imprimés assurent l'indépendance des commandes, chaque carte traitant un signal différent en provenance de la boîte de commande.

Il existe enfin le Memocon Crawler, le plus riche en possibilités.

Est-ce un robot?

Toutes les machines décrites jusqu'ici sont amusantes et intéressantes à construire, mais la technologie utilisée n'est pas très évoluée. La technologie du plus « perfectionné » de ces Movit, le Circular, existe depuis un certain temps sur des bateaux et avions télécommandés. Pour qualifier un dispositif de « robot », celui-ci doit être doté d'un système de guidage intégré qui puisse être programmé pour exécuter une séquence d'interventions.

Le Memocon Crawler est une machine qui correspond assez bien à la définition que nous avons

donnée d'un robot; comme les autres Movit, il est actionné par des moteurs CC électriques. Le système de guidage est composé d'un clavier connecté au Crawler au moyen d'un câble-ruban. Le clavier comporte des boutons séparés, correspondant à chacune des commandes disponibles : avant, gauche, droite, pause, produire un signal sonore et allumer les DEL de la machine. Ces commandes sont stockées dans une puce RAM statique dotée d'une mémoire d'une capacité de 256 multiplets, chacun étant formé de 4 bits. Ces multiplets renferment les instructions, ne sont pas adressables individuellement et nécessitent un accès séquentiel.

De ce fait, il n'est pas possible de répéter certaines étapes au sein d'une séquence, bien qu'il soit évidemment possible de répéter la séquence entière.

De nombreuses personnes ne veulent pas acheter l'un des ces robots parce qu'ils sont vendus en kit, et donc à monter. Cependant, malgré le nombre impressionnant de pièces fournies avec chaque kit, les machines sont en fait très faciles à construire et peuvent même être montées par un enfant.

Les diverses pièces des robots sont fournies emballées dans des sacs séparés clairement numérotés. Les instructions sont données sous forme de dessins « éclatés » du Movit. Chaque pièce est dessinée séparément et clairement identifiée au moyen du numéro de son sac, et des flèches indiquent de façon précise l'endroit où elle doit aller. Dans la majorité des cas, ces dessins fournissent toutes les instructions nécessaires pour le montage, mais on dispose également d'instructions écrites pour éclaircir certaines étapes pouvant prêter à confusion. Puisque les cartes de circuits imprimés sont déjà assemblées, des pinces, un tournevis et un petit marteau sont les seuls outils nécessaires.

Les Movit sont sans doute agréables à construire, et les acheteurs éprouveront une grande satisfaction en voyant évoluer au sol un engin qu'ils auront eux-mêmes construit. Mais on peut douter qu'ils représentent une véritable introduction à l'univers de la robotique. En fait, la plupart de ces dispositifs ne sont que des jouets électroniques perfectionnés.

Ils sont cependant bon marché et, bien qu'ils ne puissent pas beaucoup nous apprendre sur la robotique, ils peuvent donner à certaines personnes, particulièrement aux enfants, une bonne initiation à la robotique.

Et que ça tourne !
Le Circular, comme le reste de la gamme, est facile à construire. Les cartes de circuits imprimés du robot et de la boîte de commandes sont déjà assemblées. Cela signifie qu'aucune soudure ne doit être effectuée pendant la construction et que tout le monde peut construire son propre Movit sans posséder de notions d'électronique préalables.
(Cl. Chris Stevens.)

MONKEY	LINE TRACER II	PIPER MOUSE	CIRCULAR	MEMOCON CRAWLER
MOUVEMENT	**MOUVEMENT**	**MOUVEMENT**	**MOUVEMENT**	**MOUVEMENT**
Deux bras à mouvement alternatif à manivelle.	Trois roues actionnées par deux moteurs CC.	Trois roues actionnées par deux moteurs CC.	Deux roues actionnées par deux moteurs CC.	Trois roues actionnées par deux moteurs CC.
COMMANDE	**COMMANDE**	**COMMANDE**	**COMMANDE**	**COMMANDE**
Capteur sonore.	Capteur infrarouge.	Capteur son supersonique.	Boîte de télécommande.	Mémoire programmable.
ALIMENTATION	**ALIMENTATION**	**ALIMENTATION**	**ALIMENTATION**	**ALIMENTATION**
Deux piles de 1,5 V.	Deux piles de 1,5 V et une pile de 9 V.	Deux piles de 1,5 V et une pile de 9 V.	Trois piles de 1,5 V, une de 9 V, et une autre pour la commande.	Deux piles de 1,5 V et une pile de 9 V.

Impedimenta

Un Nouveau Monde
Les premiers grands voyages d'exploration ont eu lieu à la Renaissance, à la fin du XVe siècle. Cent ans plus tard, les navigateurs européens commençaient à peine à tracer les cartes de tous les continents. Elles ne furent d'abord que des croquis approximatifs, où îles et continents n'étaient pas toujours à la bonne place. Au fur et à mesure que notre jeu de simulation progressera, nous découvrirons un autre aspect de la carte du monde, un peu à la manière des grands explorateurs...
(Cl. Ian MacKinnell.)

Nous avons déjà l'équipage et acheté des provisions. Avant de partir pour le Nouveau Monde, il nous reste à faire l'acquisition de quelques petites choses.

Les salaires de l'équipage lui seront versés quand le navire reviendra au port (s'il y revient). Le joueur doit donc faire un choix de marchandises qu'il puisse revendre et déterminer ce qu'il doit mettre de côté, si cela lui est possible. Il peut acheter des médicaments, des armes, du sel, du tissu, des couteaux et des bijoux; à chaque fois, il se verra communiquer les prix correspondants et demander la quantité choisie. Après chaque opération, l'état de son compte est affiché; s'il n'a plus assez d'argent, il devra recommencer. Le navire peut accueillir une quantité illimitée de marchandises. Une fois achetées, le voyage peut commencer. Avant le départ, le programme affiche la composition de l'équipage, la liste des provisions et des marchandises, ainsi que le nombre de pièces d'or restantes.

Tout comme dans les deux modules précédents, les tableaux consacrés aux noms, aux prix et aux quantités des marchandises sont DIMensionnés en début de programme. Ils viennent s'ajouter aux diverses instructions d'initialisation commençant à la ligne 30. Le nombre de chaque élément de même type est géré par le tableau OA(). La description des marchandises est contenue dans le tableau D$() DIMensionné à la ligne 31, et dont les éléments sont précisés aux deux lignes suivantes. D$(1) devient FLACON DE MÉDICAMENT, et D$(2) à D$(6) deviennent ARME À FEU, SAC DE SEL, BALLOT DE TISSU, COUTEAU et BIJOU. Là encore, se référer à un objet, par l'intermédiaire de son adresse dans un tableau, facilitera la programmation.

Le prix des marchandises est DIMensionné dans un tableau OC(), qui contient six éléments aux lignes 34 et 35. Le premier élément, OC(1), représente le prix du médicament : une pièce d'or par flacon. Les armes à feu coûtent cinq pièces d'or, etc.

Au sein du programme principal, la ligne 500 appelle un sous-programme pour l'embauche de l'équipage, et la ligne 550 en appelle un autre pour l'achat des provisions. Nous allons maintenant y insérer un appel au programme d'achat de marchandises. Il va de la ligne 3000 à 3999; examinons-le en détail. L'écran est d'abord effacé; après une pause, le lignes 3005 et 3010 impriment et soulignent le titre. Le joueur apprend alors qu'il a encore besoin de diverses choses avant d'entreprendre son voyage.

L'achat des six catégories de marchandises est organisé sous forme de boucle. Celle-ci est utilisée six fois de suite (une par catégorie); T sert de compteur de boucle. Le prix de chaque article est affiché, et son nom est représenté à la ligne 3070,

par D$(T). Au début des achats, T est fixé à 1 pour le premier article, le flacon de médicament. Le prix unitaire de chaque élément est fixé à la ligne 3080 par OC(T). Ici, il sera donc égal à une pièce d'or. Au second passage à travers la boucle, T est fixé à 2, D$(T) à ARME À FEU, OC(T) à cinq pièces d'or, et ainsi de suite jusqu'à ce que (T) atteigne 6.

Formatage du texte

Si un article coûte plus d'une pièce d'or, la ligne 3081 ajoute un S à PIÈCE. Dans le cas contraire, tout reste en l'état. Vient ensuite une pause, et le programme demande DÉSIREZ-VOUS ACHETER? (O/N). La ligne 3110 attend une réponse P$, et la ligne 3120 extrait de celle-ci le caractère le plus à gauche. Si c'est un N, on passe ligne 3175, qui renvoie le programme en début de boucle, ce qui nous conduit à la marchandise suivante. Si la réponse est O, la ligne 3130 demande QUELLE QUANTITÉ VOULEZ-VOUS?

La ligne 3135 gère la réponse : vous pouvez vous contenter de taper 10, ou préciser 10 SACS DE SEL. Cela est possible parce que la ligne 3140 convertit la réponse et se sert de VAL pour reconnaître les caractères numériques du début de la phrase, en ignorant le reste.

Ne cherchez pas à finasser; si vous essayez de faire des achats interdits (sans avoir la somme nécessaire), cela sera découvert par la ligne 3145, qui s'assure que le prix de l'article, OC(T), multiplié par le nombre d'entrées, TT, n'est pas supérieur à MO, somme d'argent qui vous reste. Si c'est bel et bien le cas, on passe en ligne 3150, qui vous informera que vous manquez de fonds. La ligne 3154 invite alors à recommencer l'opération, et on en revient à la ligne 3130. Si cette fois vous avez de quoi régler vos dépenses, vous arriverez à la ligne 3160, où ce que vous venez de dépenser sera soustrait de votre cagnotte grâce à la formule MO = MO – (OC(T)*TT). Le coût de l'article, OC(T), est multiplié par le nombre d'exemplaires, TT, et le résultat est soustrait de la somme restante, MO, de façon à donner l'état en cours du solde. TT est géré par le tableau OA(), ligne 3165. Après une pause, la ligne 3176 imprime une ligne vierge et le solde de votre compte. Puis la valeur de T est incrémentée de 1, et la ligne 3200 renvoie le programme en début de boucle, de façon à passer à un nouvel article.

Une fois toutes les marchandises achetées, la ligne 3210 informe le joueur que la phase 3 est terminée. La ligne 3230 provoque l'affichage d'APPUYEZ SUR UNE TOUCHE POUR CONTINUER, en se servant de K$, variable définie en tout début de programme. Ce n'est qu'après avoir obtempéré que vous reviendrez en programme principal. Reste à donner la liste des achats et le relevé détaillé des membres d'équipage. Un sous-programme d'« ouverture » s'en chargera.

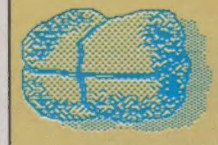
Il affiche « VOTRE ÉQUIPAGE COMPREND » et imprime la liste des hommes suivant leur catégorie en établissant une boucle (ligne 645), tandis que la ligne 650 vérifie si tel ou tel type de matelot a bien été recruté, par analyse de CC(T). Si la valeur est de zéro, la ligne 670 renvoie le programme en début de boucle, et augmente de 1 la valeur de T.

Pour toute embauche, la catégorie et le nombre de marins sont affichés, avec un S après chaque description s'il y en a plusieurs (dans le cas contraire, c'est un espace qui est imprimé). Les provisions font l'objet d'un traitement analogue.

Marchandises à part

Les marchandises ne sont pas affichées à l'aide d'une boucle, et ce, parce que leurs descriptions sont très différentes. Elles ne constituent pas des unités de mesure, et il n'est pas toujours judicieux d'ajouter un S à la fin de chaque mot (ballot devient ballots, mais couteau devient couteaux, par exemple). Il faudra donc analyser le tout séparément, de façon linéaire. Si une marchandise particulière n'a pas été achetée, le programme passe directement à la suivante. La ligne 730, par exemple, vérifie si le joueur a cru bon d'acheter des flacons de médicaments. Il se rend ligne 740 si la réponse est non. En revanche, il

De bonnes affaires
Voici les marchandises que nous emporterons avec nous pour pouvoir commercer avec les indigènes. Il faudra bien sûr les payer avant de pouvoir les emporter. Dans notre jeu, le nom de chacune d'elles est contenu dans le tableau D$; le prix de chaque article est en OC, et le nombre d'articles du même type en OA.
(Cl. Ian McKinnell.)

	(1)	(2)	(3)	(4)	(5)	(6)
D$	 Flacon de médicament	 Arme à feu	 Sac de sel	 Ballot de tissu	 Couteau	 Bijou
OC	1	5	1/5	1 1	1/2	1
OA	6	20	15	32	20	10

détermine s'il y a un ou plusieurs flacons, affiche le résultat et, suivant le cas, imprime à la suite un S ou un espace. Après quoi, il recommence l'opération sur une autre catégorie. Une fois la liste complète, la ligne 792 indique quelle est la somme d'argent qui reste et la ligne 797 vous avertit de PRESSER UNE TOUCHE POUR CONTINUER LE VOYAGE, en attendant votre réponse.

Module 3 : marchandises

Dimensionnement tableaux

```
30 DIMOA(6)
31 DIMO$(6)
32 D$(1)="FLACON DE MEDICAMENT":D$(2)="ARME":D$(3)=
"SAC DE SEL"
33 D$(4)="BALLOT DE TISSU":D$(5)="COUTEAU":D$(6)="BIJOU"
34 DIMOC(6)
35 OC(1)=1:OC(2)=5:OC(3)=.2
36 OC(4)=2:OC(5)=.5:OC(6)=1
```

Appel du sous-programme

```
600 GOSUB 3000
```

Sous-programme d'ouverture

```
605 **** PRET A PARTIR ****
610 PRINTCHR$(147)
615 S$="VOUS VOICI PRET*":GOSUB9100
625 S$="A PARTIR.*":GOSUB9100
630 GOSUB9200
635 PRINT:S$="VOTRE EQUIPAGE COMPREND:*":GOSUB
9100
640 GOSUB9200
645 FORT=1TO5
650 IFCC(T)=0THEN670
655 PRINTCC(T);
660 PRINTC$(T);
662 IF CC(T)=1THENPRINT" ":GOTO668
664 PRINT"S"
668 GOSUB9200
670 NEXT
674 GOSUB9200
675 PRINT:S$="ET VOUS AVEZ LES PROVISIONS SUIVANTES:*":GOSU
B9100
680 GOSUB9200
685 FORT=1TO4
690 IFPA(T)=0THEN710
695 PRINTPA(T);U$(T);"S DE ";
700 PRINTP$(T)
708 GOSUB9200
710 NEXT
715 GOSUB9200
720 PRINT:S$="VOUS AVEZ AUSSI:*":GOSUB9100
725 GOSUB9200
730 IFOA(1)=0THEN740
733 IFOA(1)=1THENS$="FLACON DE MEDICAMENT*":GOTO
735
734 S$="FLACONS DE MEDICAMENT*"
735 PRINTOA(1);:GOSUB9100
736 GOSUB9200
740 IFOA(2)=0THEN750
743 IFOA(2)=1THENS$="ARME":GOTO745
744 S$="ARMES"
745 PRINTOA(2);:GOSUB9100
746 GOSUB9200
750 IFOA(3)=0THEN760
753 IFOA(3)=1THENS$="SAC DE SEL*":GOTO755
754 S$="SACS DE SEL*"
755 PRINTOA(3);:GOSUB9100
756 GOSUB9200
760 IFOA(4)=0THEN770
763 IFOA(4)=1THENS$="BALLOT DE TISSU*":GOTO765
764 S$="BALLOTS DE TISSU*"
765 PRINTOA(4);:GOSUB9100
766 GOSUB9200
770 IFOA(5)=0THEN780
773 IFOA(5)=1THENS$="COUTEAU*":GOTO775
774 S$="COUTEAUX*"
775 PRINTOA(5);:GOSUB9100
776 GOSUB9200
780 IFOA(6)=0THEN790
783 IFOA(6)=1THENS$="BIJOU*":GOTO785
784 S$="BIJOUX*"
785 PRINTOA(6);:GOSUB9100
786 GOSUB9200
790 GOSUB9200
792 PRINT:PRINT"IL VOUS RESTE":MO; "DES PIECES D'OR"
796 GOSUB9200
797 S$="APPUYEZ SUR UNE TOUCHE POUR PRENDRE LA MER*"
798 GOSUB9100
799 GETP$:IF P$=""THEN799
999 END
```

Sous-programme marchandises

```
3000 REM **** PHASE 3 : MARCHANDISES ****
3001 PRINTCHR$(147): REM PHASE 3
3002 GOSUB9200
3005 PRINT"      PHASE 3 - MARCHANDISES"
3010 PRINT"      ------- -------------"
3020 GOSUB9200
3025 PRINT
3030 S$="D'AUTRES CHOSES PEUVENT SERVIR*":GOSUB91
00
3035 S$="AU COURS DU VOYAGE.*":GOSUB9100
3040 S$="AINSI MEDICAMENTS ET MARCHANDISES*":GOSUB9100
3045 S$="A ECHANGER.*":GOSUB9100
3046 GOSUB9200
3050 S$="VOUS AUREZ AUSSI BESOIN D'ARMES.*":GOSUB9100
3055 GOSUB9200:GOSUB9200
3060 FORT=1TO6
3065 PRINT
3066 W$="UN" : IF D$(T) = "ARME" THEN W$="UNE"
3070 PRINT W$ ; D$T;
3075 S$="COUTE*":GOSUB9100
3080 PRINTOC(T);
3081 PRINT"PIECE D'OR";
3085 IFOC(T)=1THENPRINT" ":GOTO3090
3086 PRINT"S"
3090 GOSUB9200
3095 S$="VOULEZ-VOUS ACHETER (O/N)?*":GOSUB9
100
3110 INPUTP$:P$=LEFT$(P$,1)
3115 IF P$<>"O" AND P$<>"N" THEN 3095
3120 IF P$="N"THEN3175
3125 GOSUB9200
3130 S$="COMBIEN?*":GOSUB9100
3135 INPUTP$
3140 TT=VAL(P$)
3145 IFOC(T)*TT>MOTHEN3150
3147 GOTO3160
3150 S$="PAS ASSEZ D'ARDENT !!*":GOSUB9
100
3152 GOSUB9200
3154 S$="RECOMMENCEZ !*":GOSUB9100
3155 GOSUB9200:GOTO3130
3160 MO=MO-(OC(T)*TT)
3165 DA(T)=TT
3170 GOSUB9200
3175 PRINT
3176 PRINT"IL VOUS RESTE =" :MO:GOSUB9100
3200 GOSUB9200:NEXT T
3205 GOSUB9200:PRINT:PRINT
3210 S$="FIN DE LA PHASE 3*":GOSUB9100
3220 GOSUB9200:PRINT
3230 S$=K$:GOSUB9100
3240 GETP$:IFP$=""THEN3240
3999 RETURN
```

Variantes de basic

Spectrum :
Procédez aux modifications suivantes :

```
 32 DIM D$ (6,20)
610 CLS
799 LET P$ = INKEY$: IF P$ ="" THEN GO TO 799
3001 CLS
3111 INPUT P$ : LET P$= P$ (1 TO 1)
3240 LET P$=INKEY$:IF P$ = "" THEN GO TO 3240
```

BBC Micro :
Procédez aux modifications suivantes :

```
610 CLS
799 A$=GET$
3001 CLS
3240 A$=GET$
```

Grand prix

Voici un jeu très connu qui reste toujours en tête des hit-parades des jeux de café. Nous vous présentons ici une version pour micro-ordinateur MSX.

Au volant de votre formule 1, essayez de parcourir la plus grande distance possible. Votre voiture dispose de deux vitesses. Maintenez la barre d'espacement enfoncée pour rester en seconde vitesse. La direction est commandée par les touches du curseur. En seconde vitesse, votre voiture roule deux fois plus vite. Mais gare à l'accident.

```
10 REM ***************
20 REM * GRAND-PRIX *
30 REM ***************
40 KEY OFF
50 WIDTH 39
60 GOSUB 450
70 H=STICK(0)
80 PX=PX+(H=7)-(H=3)
90 IF STRIG(0) THEN T=2 ELSE T=1
100 IF VPEEK(PX+40)<>CR THEN 240
110 RX=RX+(RND(1)<.5)-(RND(1)<.5)
120 IF RX<RN THEN RX=RN
130 IF RX>RM THEN RX=RM
140 VPOKE XP,CR
150 LOCATE RX,24,0
160 PRINT R$
170 VPOKE PX,V
180 K=K+T
190 DL=(2-T)*50
200 FOR I=1 TO DL
210 NEXT I
220 XP=PX
230 GOTO 70
240 VPOKE XP,CR
250 FOR I=1 TO 5
260 VPOKE PX+40,A
270 FOR J=1 TO 100
280 NEXT J
290 BEEP
300 VPOKE PX+40,V
310 FOR J=1 TO 100
320 NEXT J
330 NEXT I
340 LOCATE 10,10,0
350 PRINT "KMS PARCOURUS :";K;
360 IF INKEY$<>"" THEN 360
370 LOCATE 14,16,0
380 PRINT "UNE AUTRE ?";
390 D$=INKEY$
400 IF D$="" THEN 390
410 IF D$<>"N" AND D$<>"n" THEN RUN
420 LOCATE 0,0,1
430 CLS
440 END
450 CLS
460 SCREEN 0,0
470 GOSUB 680
480 DEFINT A-Y
490 COLOR 14,12
500 A=221
510 R$=CHR$(219)+CHR$(219)+CHR$(219)
520 CR=219
530 RX=18
540 V=128
550 RY=24
560 T=1
570 PX=420
580 XP=PX
590 LOCATE 0,0,0
600 FOR I=0 TO 24
610 LOCATE RX,I,0
620 PRINT R$
630 NEXT I
640 VPOKE PX,V
650 RM=34
660 RN=2
670 RETURN
680 FOR I=0 TO 7
690 READ A
700 VPOKE 3072+I,A*4
710 NEXT I
720 RETURN
730 DATA 18,0,18,51,51,18,0,18
```

Du français sur Apple

Nouveau traitement de texte français pour Apple IIe et Apple IIc, Gribouille est un logiciel différent. Il s'adresse aussi bien aux secrétaires, aux littéraires qu'aux ingénieurs et cadres scientifiques.

Pourquoi un nouveau logiciel de traitement de texte? Son auteur, Madeleine Hodé, s'est mise à l'informatique pour créer le traitement de texte : celui qu'elle avait envie d'utiliser.

Il se distingue d'abord par ses objectifs qui sont les différentes catégories de public auxquelles il s'adresse. Ensuite, par un certain nombre de points forts qui le placent bien au-dessus des traitements de texte français existant jusqu'à présent sur le marché.

Gribouille — c'est son nom — évoque un personnage de contes enfantins et, en même temps, l'idée de gribouillage qui rappelle très bien l'idée de traitement de texte. Son auteur l'a conçu à partir du logiciel qu'elle utilisait sur son micro-ordinateur Apple II, AppleWriter, qu'elle trouvait bon mais très imparfait.

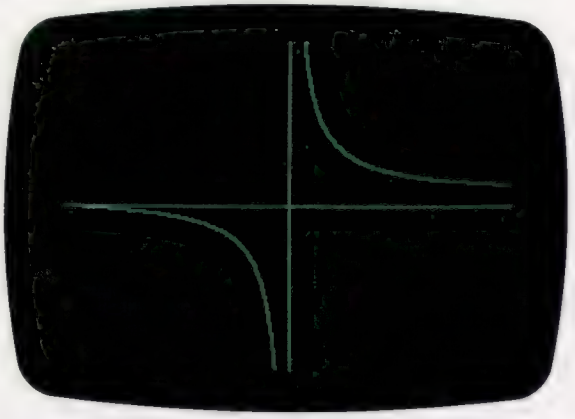
Le principal défaut des traitements de texte réalisés d'après des programmes anglais ou américains est qu'ils ne coupent pas les mots en fin de ligne. En effet, la coupure des mots anglais est si complexe qu'il existe des dictionnaires spéciaux indiquant où ces mots peuvent être coupés. En revanche, en français, la coupure se fait systématiquement entre deux syllabes, et il est donc possible de concevoir un programme pour cela.

C'est l'un des atouts de Gribouille. En outre, les bas de pages sont calculés de manière à terminer un paragraphe s'il ne reste qu'une ou deux lignes, afin de ne pas commencer une nouvelle page par une seule ligne.

Un autre élément, très utile dans ce programme, c'est les compteurs qui apparaissent à l'écran, indiquant le nombre de signes, lorsqu'on appuie sur une touche. Le nombre donné par le compteur est le nombre d'octets utilisés en mémoire. L'objectif de cette fonction « longueur du texte » est d'indiquer à l'utilisateur pendant combien de temps il peut encore écrire. Mais ce n'est pas le nombre exact de caractères qui seront imprimés car, dans certains cas, les octets en mémoire ne se traduisent pas par des caractères sur l'imprimante. Ce sont, par exemple, le retour chariot, l'accent circonflexe — qui occupe en mémoire trois caractères —, et les directives de mise en page incorporées au texte.

Le programme proprement dit occupe 36 Koctets, y compris un programme de chargement qui, ensuite, est effacé et libère la place pour le texte. Le nombre de caractères restant disponibles pour le texte et le glossaire est de 23 000, c'est-à-dire un peu moins que le texte disponible sur AppleWriter. Cela est dû au fait que le programme contient beaucoup plus de choses, donc qu'il est plus long. Il comporte, en effet, de nombreux avantages qui peuvent passer inaperçus à l'utilisateur, mais qui leur suppriment une gêne : par exemple, le programme ne va pas à la ligne avant un signe de ponctuation : il coupe le mot et rejette la fin du mot et le point à la ligne suivante.

Par ailleurs, Gribouille peut aussi fonctionner sans la carte 80 colonnes, et en conséquence sans utiliser l'extension mémoire de cette carte. Il utilise alors seulement les 64 K + 16 K de la carte langage.



Il est possible de reprendre avec Gribouille un fichier texte formaté avec AppleWriter. Les différences avec ce dernier sont les commandes de dispositifs de mise en page, caractères gras, etc. Mais Gribouille peut fort bien lire et rechercher un texte composé par AppleWriter.

Trois catégories d'utilisateurs

Gribouille s'adresse à trois catégories de public, pour lesquelles sont programmées les différentes fonctions dont ces utilisateurs ont besoin.

En premier lieu, l'auteur a conçu Gribouille pour des gens comme elle, cadres administratifs, de formation juridique, ainsi que pour tous ceux qui écrivent, notamment les secrétaires et cadres littéraires. A ceux-là, Gribouille offre la possibilité de travailler très vite afin que même une dactylo très expérimentée ne distance pas le programme; il permet aussi une correction aisée et rapide du texte. Ces corrections peuvent être de deux types :

— Les fautes de frappe, par exemple l'interversion de deux lettres. Dans ce cas, il suffit de mettre le curseur sur l'une des lettres et de taper un code; les deux lettres sont alors rétablies dans le

bon ordre sans que l'on ait à les effacer et à les réécrire. Il existe d'autres astuces de correction, permettant en général de corriger en une seule frappe. En outre, pour les accents circonflexes, aucune manœuvre spéciale n'est exigée.

— Les corrections de style constituent une autre fonction très importante des programmes de traitement de texte. Elles consistent généralement à alléger les phrases, les couper, déplacer des membres de phrases, remplacer des verbes, chercher si certains mots sont trop souvent employés... Ces déplacements et modifications s'effectuent très rapidement et commodément avec Gribouille. Pour les cadres littéraires et les secrétaires, la mise en page constitue une fonction très importante. Celle-ci est particulièrement soignée par Gribouille, du fait de la coupure des mots en fin de ligne, des paragraphes en fin de page. De plus, l'utilisateur peut choisir de ne pas couper les mots. Cette mise en page, facile à piloter, peut être visualisée à l'écran en tapant Contrôle-B. En frappe normale, le texte est affiché au kilomètre, afin de ne pas ralentir la frappe.

La deuxième catégorie d'utilisateurs à laquelle s'adresse Gribouille est constituée par les cadres de formation mathématique et les ingénieurs. A ceux-ci, elle offre une impression de graphiques préparés par d'autres programmes (du type histogrammes, etc.). Ces derniers peuvent être incorporés au texte au moment voulu en les appelant par leur nom. Gribouille permet aussi la création de caractères étrangers à l'alphabet français : symboles mathématiques ou lettres grecques, par exemple. La fonction calcul n'existe pas sur AppleWriter. Avec Gribouille, en revanche, il est possible de calculer par exemple la valeur de sin ($\pi/2$). Il est possible de calculer jusqu'à trente nombres avec n'importe quel niveau d'imbrication de parenthèses et la quasi-totalité des fonctions courantes (sin, log, arcsin, etc.).

Enfin, Gribouille est destiné aux petits commerçants. Il leur permet d'augmenter la capacité du glossaire. Celui-ci peut être enregistré et imprimé. Il va pouvoir contenir non seulement les adresses, les formules courantes de la correspondance, mais également les nomenclatures de stocks, les tarifs. La place disponible pour le texte peut alors être réduite à une ou deux pages, ce qui suffit en général pour la correspondance commerciale. Le calcul de factures n'est pas perturbé par un libellé en clair des marchandises et des conditions de facturation. Le montant est calculé instantanément. Il est utilisable directement. La facturation est automatique : le tarif est enregistré dans le glossaire. Il suffit de taper la quantité à facturer, le code du produit et la touche de calcul. La ligne de facture apparaît aussitôt. Une autre touche commande la totalisation des colonnes de nombres. La facture est ainsi complète.

A tous ces utilisateurs, Gribouille offre enfin un apprentissage ultra-facile et rapide : toutes les indications nécessaires s'affichent à l'écran. Un aide-mémoire en dix tableaux peut être appelé pendant la frappe. Il s'affiche instantanément. Enfin, le programme est accompagné d'un manuel complet, qui couvre tous les problèmes, y compris les plus complexes. Il est complété par une disquette d'exercices, indispensable pour un apprentissage concret du programme.

Pour parvenir à une telle maîtrise technique de ces différentes fonctions du traitement de texte, Madeleine Hodé est partie d'une parfaite connaissance des besoins réels des utilisateurs, quels qu'ils soient. Elle pense recevoir des « chocs en retour » : les réactions des utilisateurs de Gribouille lui suggéreront sans doute encore d'autres améliorations et développements. Faudra-t-il l'adapter pour une autre machine? Le faire passer en système d'exploitation ProDos afin d'atteindre des capacités de disque plus grandes? Étendre la place réservée au texte en utilisant systématiquement la carte 80 colonnes?...

Depuis février 1985, ce logiciel « plus que parfait » de traitement de texte est disponible chez les concessionnaires Apple au prix de vente de 1 700 F T.T.C, et distribué par la société Berlingot.

Itérations PASCAL

Nous avons vu jusqu'ici la plupart des instructions structurées en PASCAL. Nous exposons aujourd'hui les trois structures d'itérations du PASCAL qui permettent les boucles.

Nous avons déjà étudié deux des trois méthodes PASCAL d'organisation des instructions : organisation séquentielle utilisant les « parenthèses de mot » BEGIN et END, et organisation par embranchements (à choix multiples) avec IF et CASE. Le troisième type d'organisation pour instruction est l'itération ou répétition. Le langage PASCAL possède trois constructions de boucles différentes : l'instruction FOR permet de tenir un compteur d'itérations, alors que WHILE et REPEAT fonctionnent selon une condition.

L'expression booléenne, délimitée par les mots réservés WHILE et DO, est évaluée en premier. Si le résultat est vrai, l'instruction qui suit cette construction (instruction qui peut être de tout type, dont un état structuré de quelque degré de complexité) sera automatiquement répétée et exécutée aussi longtemps que l'évaluation de la condition reste vraie. Cette dernière est calculée après chaque exécution de l'instruction se trouvant dans le « corps » des constructions WHILE ou REPEAT. Cela suppose évidemment qu'un élément au moins de l'expression à évaluer soit susceptible d'être modifié par la réalisation de l'instruction de la boucle. Voici un exemple :

```
WHILE MaxInt > 1 DO
  WriteLn (« boucles en cours »)
```

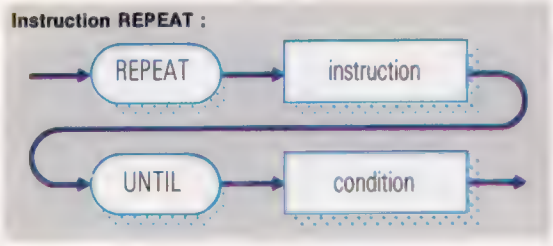
aurait du mal à trouver une fin, alors que :

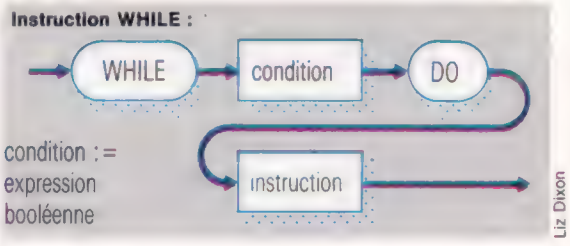
```
read (retrait) ;
WHILE retrait > solde DO
  BEGIN
    WriteLn (« solde négatif – recommencez : ») ;
    write (« Montant ? ») ;
    read (retrait)
  END
```

L'intérêt principal de la construction WHILE est illustré par l'extrait de programme précédent. Lorsque l'ordre de retrait est lu la première fois, son montant est inférieur au solde du compte, et la boucle WHILE n'est pas exécutée. Lorsque le programme parcourt la boucle, en revanche, l'expression est réévaluée après exécution de l'instruction. La boucle cessera quand l'évaluation donnera un résultat faux. L'exécution du programme reprend alors l'ordre séquentiel, la première instruction après la construction WHILE étant prise en compte. Le fait que l'expression booléenne figure au début de la syntaxe est un gage de sécurité. La sémantique du PASCAL (c'est-à-dire les nuances de sens entre les instructions ainsi que leur comportement exact) est rigoureuse. La construction WHILE est à utiliser seulement lorsque l'on désire une itération en réponse à une évaluation vraie.

Il arrive souvent que l'on ait à exécuter le corps de la boucle au moins une fois afin de réaliser la condition qui l'arrêtera, comme dans le programme du nombre d'or. La construction REPEAT... UNTIL est alors mieux appropriée.

Remarquez la condition qui termine l'itération : elle se trouve maintenant en fin de syntaxe.





Il s'agit logiquement de l'inverse de la structure WHILE. Cela signifie que la boucle REPEAT s'achève lorsque l'expression booléenne donne le résultat vrai ; la syntaxe WHILE s'arrêtant si elle est bien sûr en cours d'exécution, pour un résultat faux. Quand le corps de l'instruction d'itération REPEAT comporte plusieurs instructions, ces dernières doivent être comprises entre les mots-parenthèses BEGIN... END, en tant qu'instruction composée. Mais il se trouve que le langage PASCAL n'impose pas cette règle ici, les mots REPEAT et UNTIL faisant fonction de délimiteurs dans le corps de la boucle. Il est cependant tout à fait légal de mettre ces mots superflus, tout comme mettre deux points supplémentaires avant un mot réservé, à condition qu'une instruction nulle ne soit pas créée.

Le segment de programme suivant compte le nombre d'occurrences de la lettre « e » dans une phrase entrée au clavier. Pour obtenir l'affichage du résultat à la fin de la boucle REPEAT, terminez la phrase par un point

```
compteur : = 0;
REPEAT
```

Réponse à l'exercice du nombre d'or

Nous vous avons soumis un exercice à propos du programme du nombre d'or qui générait des termes de Fibonacci et les rapports entre les paires successives. Nous vous demandions de modifier la boucle de telle sorte que la condition finale arrête la boucle lorsque le terme de Fibonacci suivant dépassait MaxInt.

Avec la représentation du complément à deux, lorsqu'un résultat entier excède MaxInt, le bit de signe devient inintelligible. Le nombre semble négatif et le programme se plante, puisque le type de Fibonacci exclut les valeurs négatives. Même avec des entiers, le programme ne pourrait pas prendre fin. Nous ne pouvons écrire :

UNTIL premier + deuxième > MaxInt

dans la mesure où il n'existe pas par définition de nombre plus grand que MaxInt (valeur maximale pour un entier). La solution astucieuse consiste à soustraire plutôt qu'à additionner :

UNTIL deuxième > MaxInt — premier

L'arithmétique des nombres réels et la constante Epsilon ne sont alors plus nécessaires.

```
    read (symbole);
    IF symbole = « e » THEN
      compteur : = compteur + 1
  UNTIL symbole = « . »;
  WriteLn (« Il y a », compteur : 1,
    « « e » dans la phrase. »)
```

Dans le cas présent, les deux syntaxes itératives auraient pu être indifféremment utilisées, et nous aurions pu mettre en œuvre la boucle WHILE :

```
  compteur : = 0;
  read (symbole);
  WHILE symbole <> « . » DO
    BEGIN
      IF symbole = « e » THEN
        compteur : = succ (compteur);
      read (symbole)
    END;
  write (« Il y a », compteur : 1, );
  WriteLn (« e » dans la phrase. »)
```

La différence entre les deux structures est néanmoins évidente. L'événement qui détermine la condition de fin de boucle (ici, l'instruction read) se présente souvent deux fois pendant une itération WHILE, la deuxième apparaissant comme dernière instruction dans le corps de la boucle.

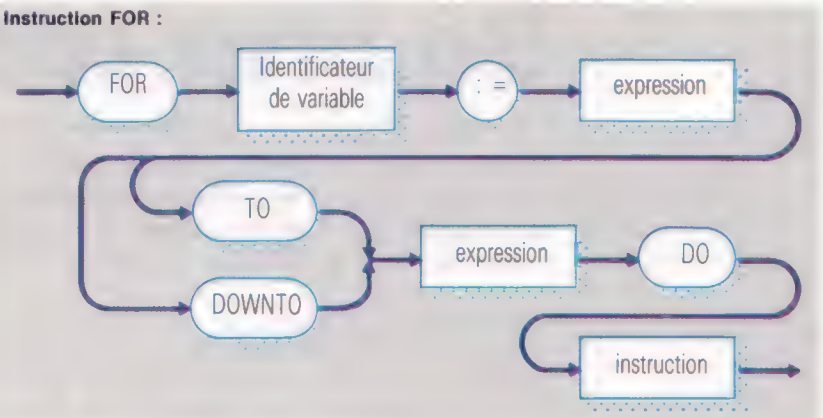
Alors que la structure WHILE constitue l'itération de base pour tout langage, il est parfois utile de pouvoir passer plusieurs fois dans une boucle. Le nombre d'itérations affecté au compteur peut être une valeur déterminée. Le PASCAL prévoit une troisième structure syntaxique pour créer de tels compteurs de boucles. Le programmeur BASIC pourra en l'occurrence se croire en terrain familier, mais la boucle FOR du PASCAL est cependant très différente. D'abord, toute variable scalaire est utilisable comme compteur, et pas seulement les entiers. Ensuite, et c'est plus important, l'instruction PASCAL FOR .. DO est tout à fait sûre. Comme la boucle WHILE, elle peut ne pas être exécutée du tout (lorsque, par exemple, la valeur de début est supérieure à celle de la fin). Il existe en outre plusieurs restrictions draconiennes à l'utilisation des variables de contrôle de la boucle. Il est notamment illégal d'en changer la valeur n'importe où dans le corps de l'itération. Et c'est même une erreur en PASCAL de l'annoncer ! Nous verrons les avantages respectifs de ces sécurités lorsque nous parlerons des procédures et des fonctions. Pour le moment, voici un exemple de syntaxe illicite :

```
FOR N : = 1 TO 10 DO
  IF N = 10 THEN
    N : = 1
```

Vous remarquez que la syntaxe de FOR .. DO comporte une instruction d'assignation entre les deux premiers délimiteurs (FOR et TO). Elle donne la valeur initiale de la variable compteur de boucles. La valeur de fin de boucle est indiquée par l'expression comprise entre les mots réservés TO et DO. Les deux valeurs limites doivent être du même type scalaire simple que celui de la variable-compteur. Voici d'autres exemples :

- FOR lettre : = « A » TO « Z » DO { etc }
- FOR mois : = Janv TO Dec DO { etc }
- FOR N : = N TO succ (MaxInt DIV 1000) DO { etc }

Instruction FOR :



Le dernier exemple suppose que l'on ait attribué préalablement une valeur à N. Souvenez-vous néanmoins que si cette dernière excède la valeur finale, la boucle ne sera pas exécutée. Dans le cas où l'itération porte sur une suite de valeurs décroissantes, le mot réservé DOWNTO est utilisé à la place de TO. Par exemple, pour un compte à rebours :

```
FOR CompteARebours : = 10 DOWNTO 0 DO
WriteLn (Compte à rebours : 32 - 3 * CompteARebours);
WriteLn (« Décollage ! »)
```

Programme « Au Fût »

```
PROGRAMME  "Au Fût"  ( sorties );

TYPE
   bière  =  ( blonde,  rousse,  brune,  spéciale );
VAR
   prix,
   litres  : réel;
   fût  : bière;
BEGIN
   WriteLn ( Litres  : 6,
      'blonde' :9,  'rousses' :8,
         'brune' :8, 'spéciale' : 8);
   WriteLn
litres := 0.5;  {commencez à 1/2 litre}

   REPEAT
     IF litres - trunc ( litres ) > 0.4
          THEN  {écrire sous la forme  ##.#}
            write ( litres : 6 : 1,  ' ' )
          ELSE  {écrire sous la forme d'un entier}
            write ( arrondir (  litres )  :4,  '  '

            :3 );


     FOR fût := blonde TO spéciale DO
         BEGIN
           CASE fût OF
              blonde   : prix := 65;
              rousse   : prix := 69;
              brune    : prix := 74;
              spéciale : prix := 79
           END;   CASE
             {arrondir et convertir en francs :}
           write ( arrondir ( prix * litres )
                                / 100 : 8 : 2 )
         END;
           {commencez une nouvelle ligne}
      WriteLn;
      litres := litres + 0.5

   UNTIL litres > 10
END  .
```

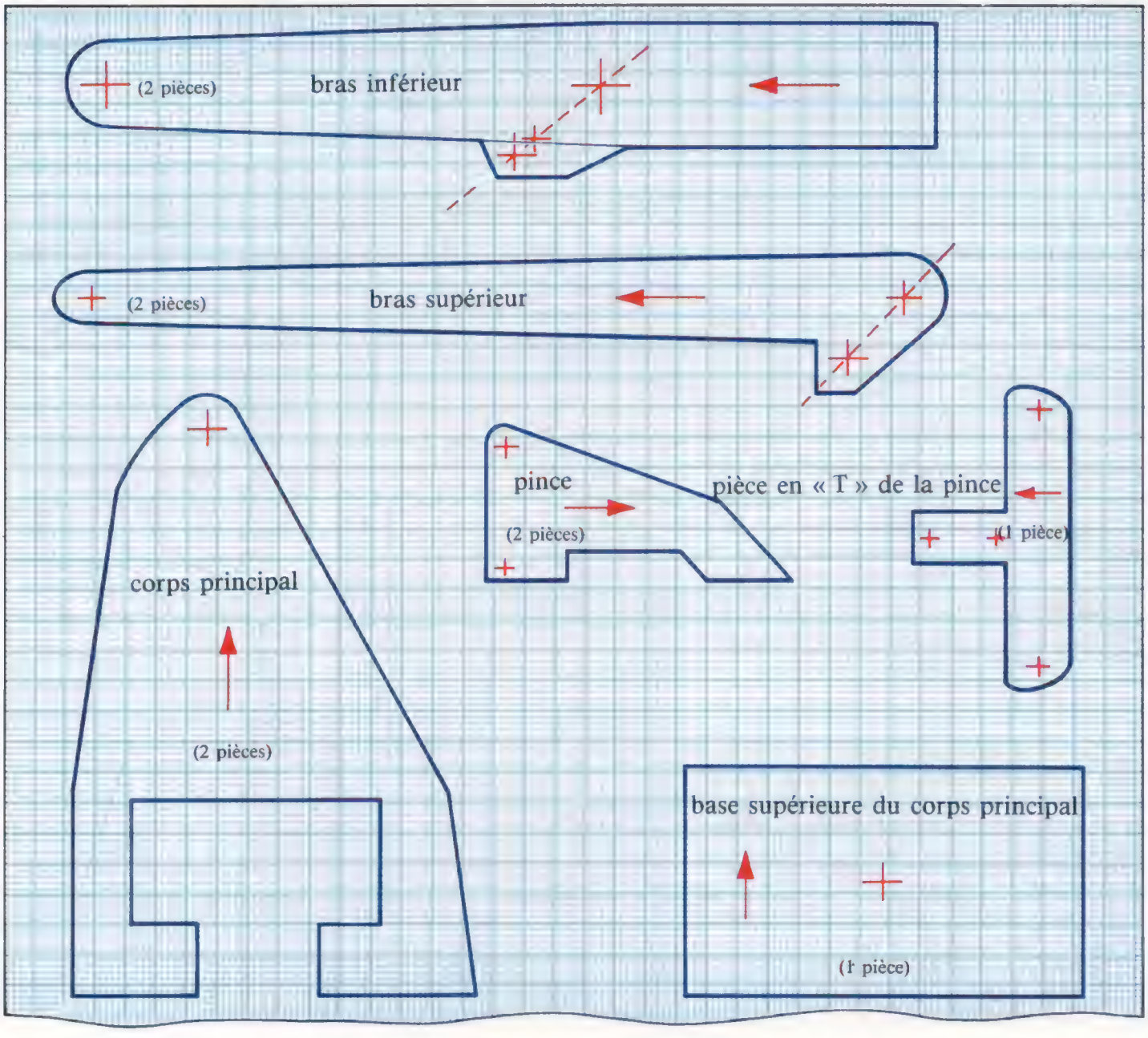
Que buvez-vous ?
Le programme « Au Fût » affiche les prix de quatre bières pour des quantités croissantes (1/4, 1/2, 3/4 et 4/4). Chacune des bières est représentée par un identificateur de constante arbitraire. Les quatre types sont donnés dans la définition TYPE de la bière. Le choix du prix associé a lieu par CASE, ce qui est tout indiqué. Un litre correspond à une valeur entière suivie de deux espaces. Par ailleurs, le formatage des quantités en tant que réels permet de supprimer les positions décimales non voulues.

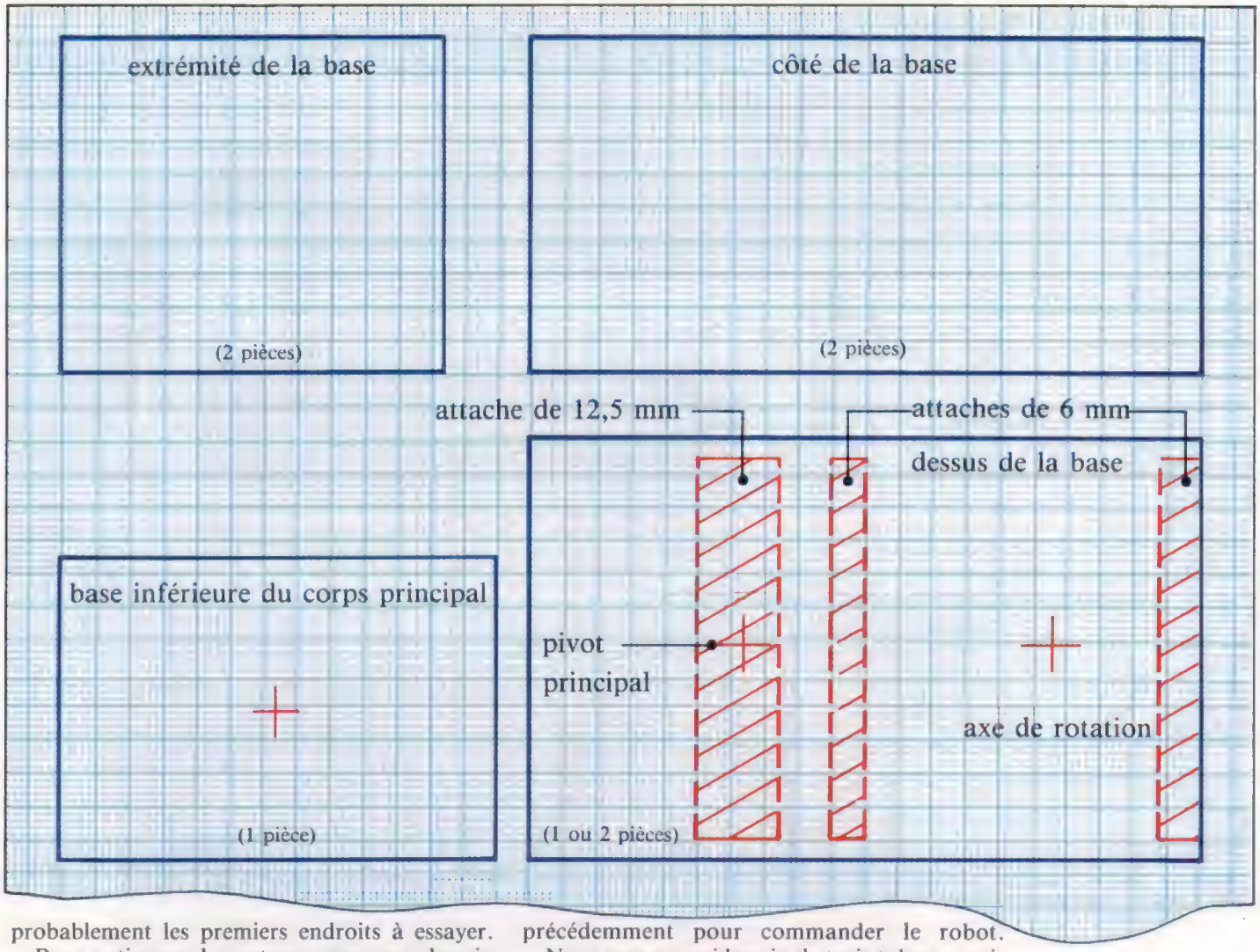
Un bras robuste

Nous commençons la construction du bras-robot en donnant d'abord les « patrons » des divers éléments du bras, puis en décrivant en détail les autres pièces nécessaires à la réalisation de ce projet.

La base et les éléments du bras doivent être construits avec un matériau léger, rigide et facile à couper. Le matériau le plus facile à se procurer est sans doute le contre-plaqué, mais il est possible d'utiliser du plastique rigide. En supposant que le contre-plaqué serve à construire l'ossature, nous aurons besoin d'une planche de 50 cm sur 50 cm, d'une épaisseur de 4 mm. Nous utiliserons ce morceau pour découper les pièces de l'ossature. Nous aurons aussi besoin de trois morceaux de bois de 55 mm de longueur (deux d'une épaisseur de 6 mm et un de 12,5 mm) qui serviront d'attaches.

Vous avez besoin de quatre servomoteurs de 5 V pour le bras. Vous devrez également vous procurer des disques plastiques de 30 mm, qui seront placés à l'extrémité de l'axe de rotation du moteur. Les boutiques de modélisme sont





probablement les premiers endroits à essayer.

Pour actionner le moteur, vous avez besoin d'une alimentation de 5 V CC. Bien que les trois micros que nous utiliserons pour commander le bras sont dotés d'alimentation de 5 V sur le port utilisateur ou sur le port d'extension, ces tensions sont conçues pour piloter des circuits électroniques externes à des niveaux logiques transistor-transistor. Le courant maximal que ces sources de 5 V peuvent fournir est de 100 mA. Puisqu'un servomoteur à pleine charge peut appeler 200 mA, il n'est pas recommandé d'alimenter les servomoteurs à partir du port utilisateur. Nous avons donc besoin d'une source d'alimentation externe. Un transformateur CC de 5 V serait idéal mais l'utilisation d'une batterie de trois piles de 1,5 V représente une solution plus économique. La sortie résultante de 4,5 V sera suffisante pour faire fonctionner le système.

Nous avons besoin d'une carte de montage de 9 trous par 20 bandes pour monter des prises où seront branchés les moteurs, et d'un câble-ruban à 20 voies de 2 m de longueur pour connecter le bras à l'ordinateur.

Les propriétaires d'ordinateurs Commodore 64 doivent se procurer un connecteur plat à 24 voies de 0,15 pouce pour connecter le bras au port utilisateur. Le bras sera relié au Spectrum au moyen de l'interface que nous avons construite précédemment pour commander le robot.

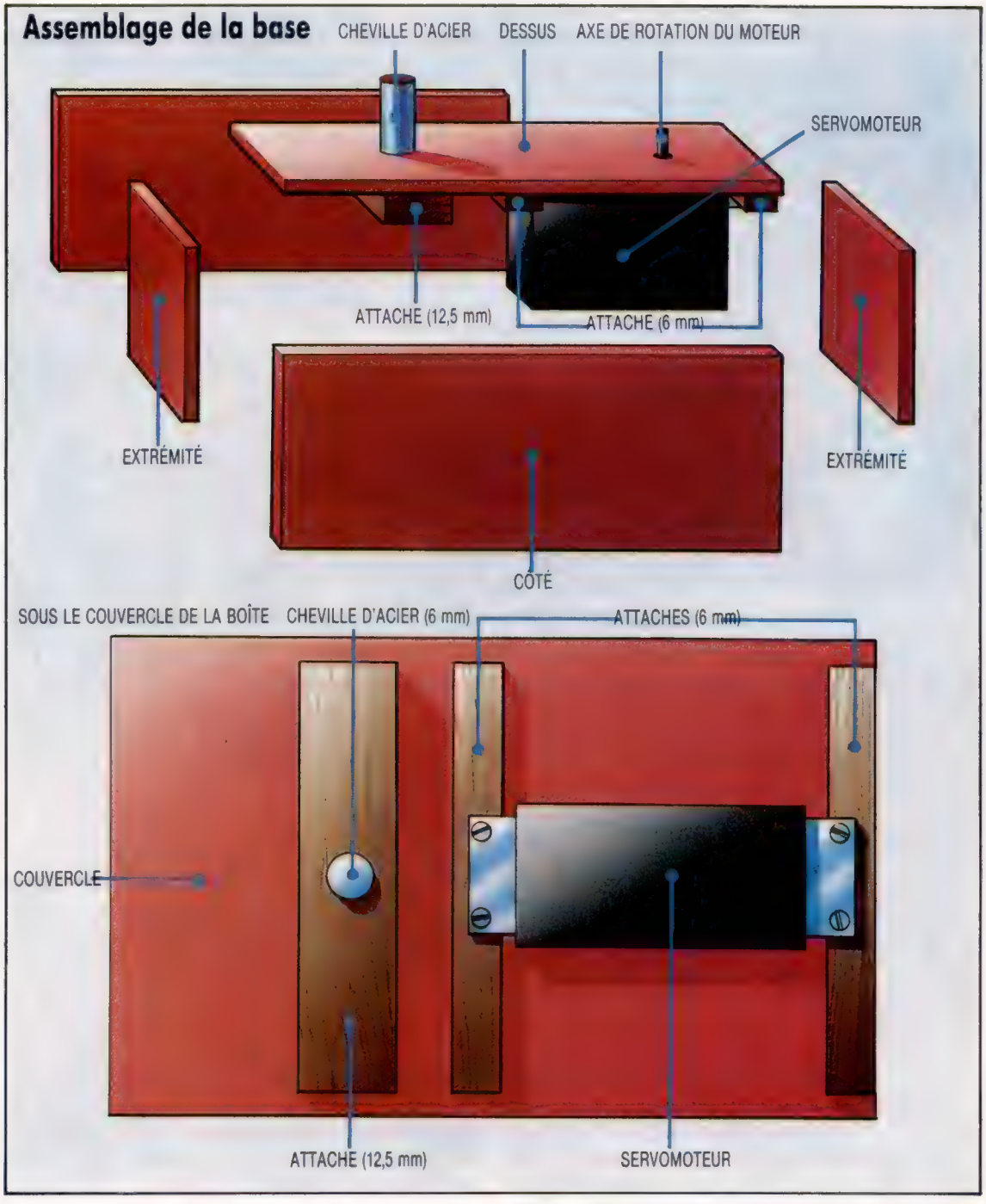
Nous avons aussi besoin de trois tubes en cuivre de 75 mm de longueur, d'un diamètre extérieur de 4 mm, ainsi que de trois autres cubes d'un diamètre de 5 mm pour construire les pivots des bras. Chaque pivot sera construit en plaçant le tube de 4 mm à l'intérieur du tube de 5 mm et en coupant les tubes aux dimensions requises. Il est donc important que les tubes que vous achetez s'emboîtent correctement, mais avec un dégagement suffisant pour permettre une libre rotation du tube intérieur.

L'ensemble du bras principal est relié à la base au moyen d'un pivot à roulement à billes. Le roulement à billes doit être bordé et doit avoir un diamètre interne de 6 mm. On utilise souvent ce type sur les roues avant des modèles réduits de

Découpe
Tracez les « patrons » de l'ossature, collez-les sur le contre-plaqué et commencez à les découper. Notez que certains patrons sont utilisés deux fois. Mettez les pièces jumelles l'une contre l'autre avant d'effectuer les perforations. Poncez soigneusement chaque pièce.

Liste des pièces

- Plusieurs vis-machine 8 BA et plusieurs boulons pour assembler la pince
- Cale d'un diamètre interne de 5 mm et 6 mm
- Colle à bois et résine époxy
- 16 vis à tôle de 10 mm et 16 rondelles de caoutchouc pour monter les moteurs
- 5 cm de mousse pour les mâchoires de la pince
- Ruban isolant



Kevin Jones

voitures. Pour allonger le palier, nous avons aussi besoin d'une cheville de 25 mm de longueur, d'une épaisseur de 6 mm. Cette cheville doit être fixée solidement à la base et doit être parfaitement proportionnée au roulement à billes.

La pince est actionnée au moyen d'un moteur monté au bas de l'ensemble du bras. Le moteur doit être connecté à la pince à l'aide d'un long câble souple. On utilise souvent ce type de câble pour commander les volets des modèles réduits d'avions. Un fil rigide de 50 mm de longueur (une corde de piano est idéale) sera nécessaire pour ouvrir et fermer la pince; il doit être relié à l'extrémité du câble souple au moyen d'un fil de fusible de 5 A. Les éléments du bras sont actionnés au moyen d'un système de tiges-poussoirs.

Construction de la base

Commencez par coller l'attache de 12,5 mm sous le couvercle de la boîte. Le trou du pivot principal doit être percé afin de loger la cheville de 6 mm. Percez le couvercle et l'attache, insérez la cheville et collez-la. Collez les deux autres attaches sur le couvercle. Ces attaches servent à monter l'un des servomoteurs. Percez l'autre trou du couvercle à la dimension exacte de l'axe de rotation, et vissez le moteur sur les attaches en utilisant les quatre vis à tôle et les rondelles de caoutchouc. Les côtés et les extrémités du couvercle peuvent être collés.

Quatre tiges métalliques de 150 mm de longueur, d'un diamètre de 2 mm, doivent être utilisées à cet effet. Les tiges peuvent être connectées aux disques des moteurs, directement ou avec des joints à rotule.

Organisation des données

Une base de données bien structurée permet d'organiser l'information. Voyons ici les idées qui sont à l'origine de la mise en place de tout système efficace d'organisation de données.

Pour le programmeur, les données peuvent être soit structurées, soit non structurées. Les données non structurées sont isolées, généralement situées dans des variables ou des constantes. On peut donner les exemples suivants : NOMBRE_DE_COUPS: =NOMBRE_DE_COUPS+1 ou IF TIR = 1 THEN PROCÉDURE INCRÉMENTATION.

Dans ces deux exemples, NOMBRE_DE_COUPS et TIR sont des variables. Dans le premier cas, la valeur de la variable est modifiée et dans le second, testée (avec un branchement possible sur une procédure). Il est manifeste que les « objets » NOMBRE_DE_COUPS et TIR ne peuvent prendre qu'une valeur à la fois.

Les programmeurs, et, surtout, ceux habitués au BASIC s'accommodent de ces données non structurées dans ces termes. Dans la vie courante, cependant, les données structurées nous sont familières et naturelles.

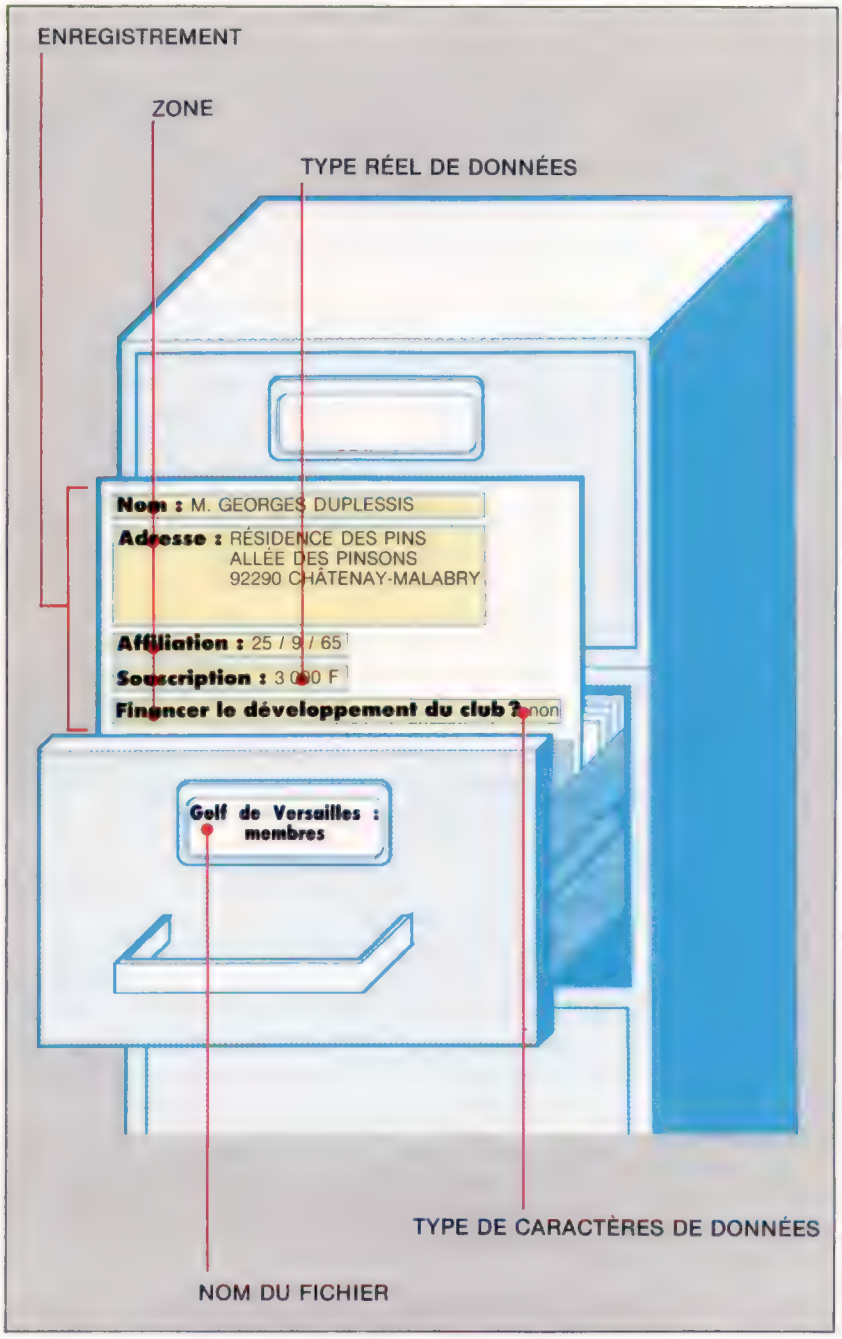
Prenons pour exemple la façon dont vous pensez à vos amis en général. Vous connaissez leur nom, leur âge (approximativement), leur sexe (difficile de se tromper...). Vous savez à peu près ce qu'ils font dans la vie, leur niveau de vie, et autres choses encore.

Nous appellerons un ensemble de faits de cette sorte ayant un dénominateur commun (une personne, un objet, une idée) un enregistrement. Chaque enregistrement consiste en une ou plusieurs zones contenant un même type de données ou des types différents. L'ensemble des enregistrements s'appelle un fichier.

Un club de golf

Pour prendre un exemple concret, considérons le secrétariat d'un club de golf devant gérer 500 adhérents. Pour en assurer la mise à jour, un index est tenu, à raison d'une fiche par membre. Chaque fiche comprend diverses informations (types de données) sur la personne concernée. Ces rubriques peuvent être, par exemple, NOM, PRÉNOM, SEXE, ÂGE, ANNÉE D'AFFILIATION, PAIEMENT DE LA COTISATION?, RESSOURCES, NUMÉRO DE MEMBRE, etc.

En langage informatique, chacun de ces éléments d'information dans un enregistrement s'appelle « zone » et les diverses zones contiennent divers types de données. NOM et PRÉNOM seront bien sûr des chaînes de caractères, ce qui veut dire que 137 4662 ne serait pas accepté comme donnée pour le nom d'un adhérent. La saisie de la rubrique SEXE supposera une variable booléenne ne prenant que deux valeurs possibles. L'ÂGE sera une valeur entière (de préférence!). ANNÉE D'AFFILIATION sera également une valeur entière prise dans une fourchette donnée. Nous supposerons qu'ANNÉE D'AFFILIATION = 1515 (Marignan?) est impossible, même si le fichier com-



Ian McKinnell

porte les enregistrements des anciens membres... De même, ANNEE D'AFFILIATION = 2001 ne sera pas admis, à moins de prendre les inscriptions pour les enfants. REVENUS sera une donnée de type réel, par exemple 150 000 F par an.

Les programmeurs doivent avoir conscience des types de données que permet le langage de programmation choisi. Certains langages, dont le BASIC et le C, sont pauvres en types de données; d'autres, comme le PASCAL, supposent que tous les éléments de données correspondent à des types strictement définis.

Si vous ne voyez pas bien l'importance de cette distinction, voyons d'abord comment le BASIC et le PASCAL traitent les données. Le BASIC ne reconnaît que deux structures possibles, les variables et les fichiers. Si le programmeur a besoin d'une autre structure, il devra la créer. Les variables BASIC peuvent appartenir à plusieurs types, dont le type entier (SCORE% par exemple), le type réel en simple précision (REVENUS DE DUPONT = 123456,66666666666 par exemple), et les variables chaînes (ADMIRABLE = 'ABCInformatique'). Le BASIC ne comporte pas le type constante, tel que PI = 3.1411592, même si vous pouvez assigner cette valeur à la variable PI. Rien ne vous empêche cependant de l'effacer en donnant un contre-ordre plus loin dans le programme, PI = 6.2 (!).

D'autres langages de programmation proposent plusieurs types prédéfinis de données, et permettent aux programmeurs d'en définir d'autres dans les mêmes termes de ceux qui existent déjà. Les types prédéfinis en PASCAL sont les constantes (les non-variables), et les variables de type « char » (caractères), « integer » (entier), « real » (pour des nombres réels tels que 12,71), et « boolean » (donnée binaire prenant deux valeurs, vraie ou fausses). Les ensembles, les tableaux, les enregistrements et les fichiers sont incorporés. Voici comment vous pourriez créer le type de données JOUR :

```
TYPE
  JOUR = (LUNDI, MARDI, MERCREDI, JEUDI, VENDREDI,
    SAMEDI, DIMANCHE);
```

Toute variable utilisée dans le programme faisant intervenir le type JOUR devra prendre sa valeur parmi celles-ci. Supposons que l'on définisse JOUR_NON_OUVRABLE selon le type JOUR;JOUR_NON_OUVRABLE := NOVEMBRE serait illégal, mais JOUR_NON_OUVRABLE := DIMANCHE serait exact.

Du point de vue du programmeur, des définitions de types très structurées comportent des avantages mais aussi des inconvénients. Passons maintenant à l'application des structures de données aux bases.

Qu'est-ce qu'une base de données?

Une base de données est un ensemble quelconque de données structurées ayant un commun dénominateur. De manière conventionnelle, les données seront organisées sous la forme d'un fichier (avec son propre nom). Ce dernier comportera des enregistrements (identifiés par un numéro). Les enregistrements comportent plusieurs zones de données, elles-mêmes constituées de plusieurs types de données.

Supposons que la base de données soit organisée à la manière d'un répertoire se trouvant dans une boîte à chaussures (vieux système) : la clef principale d'accès à chaque fiche sera certainement le nom, classé selon l'ordre alphabétique. Si vous cherchez un individu appelé Sigisbert, vous devrez feuilleter du doigt le fichier jusqu'à trouver les noms commençant par « S ». Vous ralentirez alors jusqu'à trouver la fiche. Pour un fichier relativement petit, pas de problème. Mais si vous devez établir la proportion de femmes dans votre population-fichier, la tâche deviendra fastidieuse...

Gestionnaires de bases de données

Les progiciels de gestion de bases de données ont des performances très diverses. Les plus simples sont des agendas électroniques élémentaires susceptibles de restituer un enregistrement à partir d'un seul paramètre tel que NOM = ?. Les plus compliqués comportent leur propre langage de programmation qui autorise un traitement complexe des informations de la base de données. Ces derniers permettent même de transférer les informations sur d'autres programmes (de paie, de facturation ou de traitement de texte). Nous parlerons dans cette série d'articles de toute la gamme des gestionnaires de bases de données, depuis le plus simple et attrayant, jusqu'au plus performant. Nous vous apprendrons à les utiliser au mieux.

Auparavant, reprenons la question des types de données : un bon gestionnaire devra permettre à l'utilisateur de définir le type de données à attribuer à chaque zone. Il refusera des saisies telles que NOM = 143326 ou ANNÉE D'AFFILIATION = 1066. Il devra également vous signaler comme étant une erreur des recherches du genre NOM (HOMME OU FEMME) ET REVENUS < 0. Les bons gestionnaires de bases de données, contrairement aux langages de programmation, doivent disposer de moyens solides de vérification des données. Cela évite toute saisie erronée lors de la création ou de la modification des enregistrements.

Nous verrons ultérieurement comment les divers gestionnaires organisent leurs enregistrements. Nous étudierons les bases relationnelles et multifichiers, les zones et enregistrements de longueur fixe, par opposition à ceux de longueur variable, la création des bases de données et les recherches d'informations.

Pour illustrer notre propos, nous parlerons de trois progiciels : Archive de Psion, dBase II d'Ashton Tate et Caxton's Software Card Box. Les exemples de programmation utiliseront la syntaxe des produits (logiciels) concernés. Par contre, les exemples d'ordre général seront écrits en « BASCAL », pseudo-langage hybride entre le PASCAL et le BASIC.

Trois possibilités

Le symbole de décision ou de comparaison détermine si une affirmation est vraie ou fausse, mais il peut aussi donner en outre une troisième sortie.

Le programme suivant détermine si une année quelconque entrée par clavier est bissextile (supposant, dans un premier temps, que le nombre correspondant à une telle année est multiple de 4).

Le diagramme 1 montre, pas à pas, ce qui a été d'abord traduit dans le programme 1 ; pour sa part, le programme 2 a été élaboré en suivant la même logique, mais en essayant de simplifier au maximum les opérations.

Jusqu'à présent, à chaque fois où il fut nécessaire de prendre une décision en confrontant des éléments, la comparaison a été faite uniquement en raisonnant sur le « vrai » et le « faux ».

PROGRAMME 1

```
10 REM*********BISSEXTILE
15 INPUT "ENTRER UNE ANNÉE"
20 E = INT (A/4)
30 M = E*4
40 IF M = A THEN GOTO 60
50 PRINT "L'ANNÉE N'EST PAS BISSEXTILE"
60 PRINT "EST BISSEXTILE"
70 END
```

PROGRAMME 2

```
10 REM*********BISSEXTILE
15 INPUT "ENTRER UNE ANNÉE"
20 IF A/4 = INT (A/4) THEN PRINT "EST BISSEXTILE"
30 PRINT N'EST PAS BISSEXTILE"
40 END
```

Diagramme 1

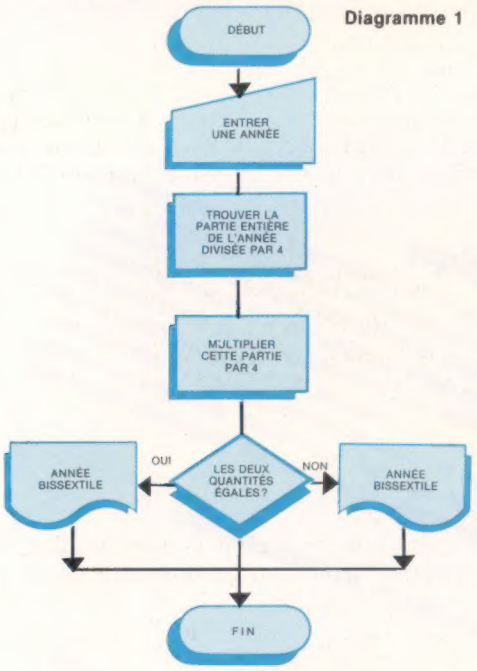
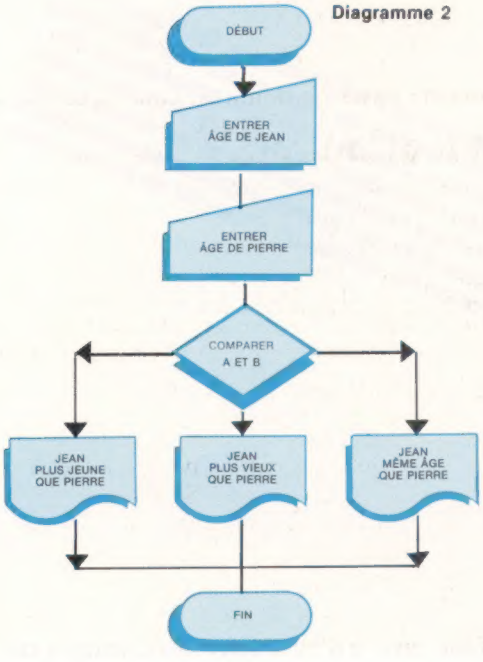
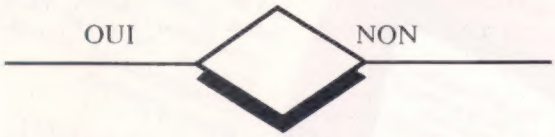


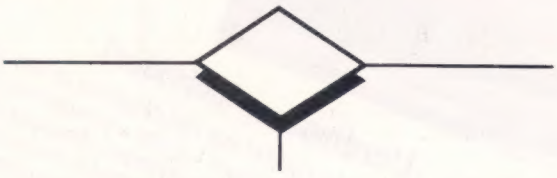
Diagramme 2



En ce sens, le losange donnait deux sorties possibles. Comme ce même symbole peut offrir une troisième voie de sortie, pourquoi s'en priver en cas de besoin? Il est permis de saisir l'opportu-



nité de distinguer, outre les deux possibilités précédentes, c'est-à-dire les sorties « inférieur » (<) ou « supérieur » (>), la sortie « égal » (=).



Dans le diagramme 2, on peut observer comment est utilisée cette technique en comparant l'âge de deux personnes. Comme on peut le constater, la séquence suivra l'une des trois sorties possibles. Ainsi, le résultat acquiert une précision supplémentaire qu'il n'aurait pu obtenir si l'on avait seulement posé la question : est-ce que Juan est plus âgé que Pablo? Car, dans le cas où la réponse aurait été négative, le doute persisterait dans la mesure où cette direction ne permettrait pas de distinguer si l'un est bien plus jeune ou s'il a le même âge que l'autre.

Livres pour débuter

Voici une sélection de livres pour vous initier à ces étranges et merveilleuses machines que sont les micro-ordinateurs. Ils vous feront entrer dans le cercle des amateurs éclairés, puis des passionnés de micro-informatique.

Initiation pratique à l'informatique

« Plus de cent exercices pour découvrir le traitement de l'information et apprendre à programmer méthodiquement. » C'est ce que vous propose, en quatre volumes, les auteurs d'*Initiation pratique à l'informatique*. Le premier volume comprend deux parties : la première présente, en six dossiers, les concepts de base de l'informatique; la deuxième est plus spécifiquement consacrée, en dix dossiers, à la programmation. Le deuxième volume se décompose en trois grandes parties : l'emploi des fichiers, l'analyse et des études de cas. Les deux derniers volumes correspondent aux corrigés de tous les exercices proposés dans les deux premiers.

Par J.L. Charron — E. Fanouillet — M. Yelloz-Dantiez.
Nathan-Technique.

L'ordinateur individuel

Un bref rappel sur les origines et les premiers pas de l'informatique. Quelques mots sur les premiers « gros » systèmes de l'informatique de papa et sur leur évolution jusqu'aux petits systèmes d'aujourd'hui. Puis on aborde l'initiation proprement dite : la rapidité et la « stupidité » intrinsèque de l'ordinateur. Suivent des notions de langage binaire et de programmation. L'interface homme/machine et les divers langages sont ensuite exposés. L'auteur présente enfin les grands organismes utilisateurs. En résumé, un excellent ouvrage d'initiation ou, mieux, un « manuel de survie » pour le profane.

Par Yves Leclerc.
Collection Initiation.
280 pages. Format 15 × 22,5 cm.
P.S.I.

Page manquante
(publicité)

Page manquante
(publicité)